CSE 4/546: Reinforcement Learning
Fall 2021

Instructor: Alina Vereshchaka

Assignment 3 - Actor-Critic

Team Registration: November 12, Fri, 11:59pm

Due Date: November 28, Sun, 11:59pm

1 Assignment Overview

The goal of the assignment is to help you understand the concept of policy gradient algorithms and to allow you to implement the actor-critic algorithm and apply it to solve OpenAI gym environments. We will train our networks on a reinforcement learning environment among OpenAI Gym or other complex environments.

Part 1 [Total: 60 points] - Implementing Actor-Critic

1. Implement actor-critic algorithm. It can be completed in any of the following: Q Actor-Critic, TD Actor-Critic, Advantage Actor-Critic (A2C), Proximal Policy Optimisation, etc. You may use any framework (Keras/Tensorflow/Pytorch).

2. Train your implemented algorithm on any environment (e.g. your grid world from A1/A2, OpenAI Gym, etc). Show and discuss your results after applying your Actor-Critic implementation on the environment. Plots should include the total reward per episode.

3. Provide the evaluation results. Run your environment for at least 10 episodes, where the agent chooses only greedy actions from the learnt policy. Plot should include the total reward per episode.

Part 2 [Total: 40 points] - Solving Complex Environments

Test your Actor-Critic algorithm implemented in Part 1 on any other TWO complex environments. You may use your custom made multi-agent environment or any other complex environment that you will use for your Final Project (this has to be approved by the course staff). You can adjust your NN structure or hyperparameters from your base implementation. Describe the environments that you used (e.g. possible actions, states, agent, goal, rewards, etc). Provide reward dynamics (average reward in t-steps). The environment with multiple versions will be considered one environment.

Suggested environments to work with:

- OpenAI CartPole
- OpenAI LunarLander
- OpenAI MountainCar
- OpenAI HandManipulateBlock

- PyBullet Robotics Environments (Documentation)
- RL Bench (link)

In your report:

1. Discuss the algorithm you implemented.
2. What is the main difference between the actor-critic and value based approximation algorithms?
3. Briefly describe THREE environments that you used (e.g. possible actions, states, agent, goal, rewards, etc). You can reuse related parts from your Assignment 2 report.
4. Show and discuss your results after training your Actor-Critic agent on each environment. Plots should include the reward per episode for THREE environments. Compare how the same algorithm behaves on different environments while training.
5. Provide the evaluation results for each environments that you used. Run your environments for at least 10 episodes, where the agent chooses only greedy actions from the learnt policy. Plot should include the total reward per episode.
6. If you are working in a team of two people, we expect equal contribution for the assignment. Provide contribution summary by each team member.

Extra Points [max +10 points]

- **Implement more complex actor-critic [5 points]**
  Extend your actor-critic algorithm to a more advanced version, e.g. TRPO/DDPG/TD3/SAC. Compare the results after applying it to the same THREE environments used in the assignment. Provide three rewards dynamic plots for each environment with the results of two algorithms: actor-critic and improved version. Disscuss the results.
- **Solve Image-based Environment [5 points]**
  Use one of the environments with image representation of the state that requires a utilization of CNN (Convolution Neural Network) for the state preprocessing (e.g. OpenAI Breakout).

2 Deliverables

There are two parts to your submission:

2.1 Report

The report should be delivered as a separate pdf file, and it is recommended for you to use the NIPS template to structure your report. You may include comments in the Jupyter Notebook, however you will need to duplicate the results in the separate pdf file. For the final submission, combine the reports for both Part 1 and Part 2 into one file.

2.2 Code

Python is the only code accepted for this project. You can submit the code in Jupyter Notebook or Python script. You can submit multiple files, but they all need to have a clear name. After executing command python main.py in the first level directory or Jupyter Notebook, it should generate all the results and plots you used in your report and should be able to be printed out in a clear manner. Additionally you can submit the trained parameters, so that the grader can fully replicate your results. For the final submission you can combine the code from both parts into one.

3 References

- NIPS Styles (docx, tex)
- Overleaf (LaTex based online document generator) - a free tool for creating professional reports
- GYM environments
- Lecture slides

4 Final Submission [Due date: November 28]

Add your combined pdf and ipynb/python script for Part 1 and Part 2 to a zip file $TEAMMATE1\_TEAMMATE2\_assignment3.zip$ (e.g. $avereshc\_nitinvis\_assignment3.zip$) and upload it to UBlearns using group submission (Assignments section). After the assignment is graded, you may be asked to demonstrate it to the instructor if your results or reasoning in your report are not clear enough.

5 Important Information

This assignment can be completed in groups of two or individually. Teams can not be the same for A2 & A3 assignments. The standing policy of the Department is that all students involved in any academic integrity violation (e.g. plagiarism in any way, shape, or form) will receive an F grade for the course. The catalog describes plagiarism as "Copying or receiving material from any source and submitting that material as one's own, without acknowledging and citing the particular debts to the source, or in any other manner representing the work of another as one's own.". Updating the hyperparameters or modifying the existing code is not part of the assignment's requirements and will result in a zero. Please refer to the UB Academic Integrity Policy.

6 Late Days Policy

You can use up to 5 late days throughout the course toward any assignments' checkpoint or final submission. You don't have to inform the instructor, as the late submission will be tracked in UBlearns. If you work in teams the late days used will be subtracted from both partners. E.g. you have 4 late days and your partner has 3 days left. If you submit one day after the due date, you will have 3 days and your partner will have 2 days left.

7 Important Dates

November 12, Fri, 11:59pm - Register your team (UBlearns > Tools > Groups)

November 28, Sun, 11:59pm - Assignment 3 is Due